From Data to Insight

Data Visualization with `ggplot2`
Coffee, Cookies, and Coding (C-cubed) Workshops

November 24, 2025

This workshop includes two handouts:

1. A codebase with starting code and solutions for discussion and challenge questions. Download and initialize it from the Accessing the Codespaces section.
2. A webpage displaying all the code for the Worked-Through Example plots.

All workshop materials, including annotated slides and usage instructions, are available on the session's Book of Workshop webpage: https://ysph-dsde.github.io/Book-of-Workshops/Workshops/Data-Visualization-with-ggplot2/visualization-index.html.

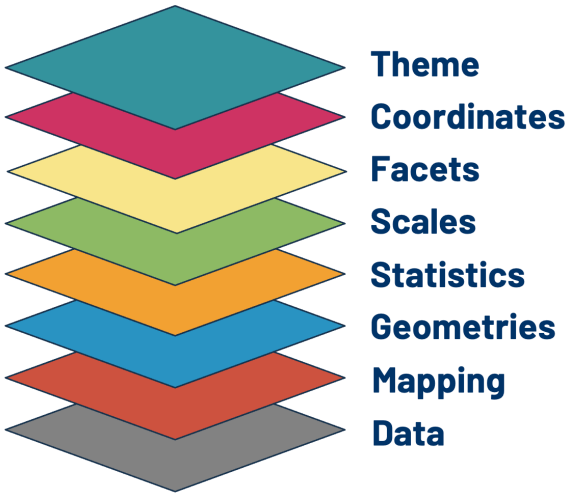

Figure 1: Figure from the *ggplot2 workshop part 1* by Thomas Lin Pedersen. Accessed from YouTube March 15th, 2025.

The Grammar of Graphics is the standard domain language for describing the components of graphical representations of quantitative information. It was first formalized by Professor Leland Wilkinson, a Yale alumnus, in his book *The Grammar of Graphics*, published in 1999 [1].

Professor Hadley Wickham adapted this framework into a layered system, which he described in his 2010 article *A Layered Grammar of Graphics*, published in the *Journal of Computational and Graphical Statistics* [2]. The figure shown on the left, created by one of the `ggplot2` developers, Thomas Lin Pedersen, illustrates the different objects that constitute a complete graph [3].

The following sections summarize key takeaways for each layer, along with the code used in the slide decks [3–9].

Data

Specify the **Data** to be plotted, ensuring it is in a tidy format, meaning that:

- Each variable is in its own column. Each column represents one variable.
- Each row represents a single observation with all associated metadata. Each observation is reduced to one row.
- Each cell contains a single value. Each value is contained in one cell.

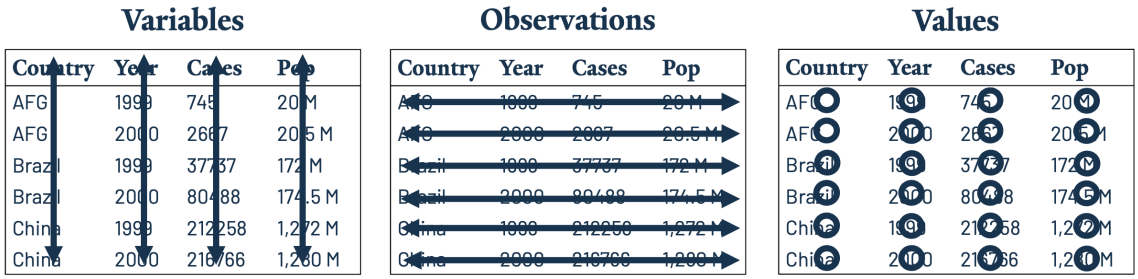

Figure 2: *R for Data Science (2e) - Introduction Figure 5.1* by Hadley Wickham, Mine Çetinkaya-Rundel, and Garrett Grolemund. Accessed March 16th, 2025.

As displayed below, our respiratory infections dataset is already in a tidy format.

Table 1: Select columns and rows from the RSV-NET dataset.

Region	Season	Week Observed	MMWRweek	Level	Positives Detected	Spline	Kernel	Crude Rate
Georgia	2018-19	2018-11-10	19	18-49 Years	3	8.68	12.51	0.1
Georgia	2019-20	2020-04-04	40	Male	3	0.95	0.81	0.1
Michigan	2024-25	2024-07-27	4	Female	0	0.00	0.00	0.0
Oregon	2021-22	2021-08-07	5	American Indian or Alaska Native	0	2.49	0.00	0.0
Oregon	2019-20	2019-12-28	26	White	25	46.73	49.54	7.8
Utah	2021-22	2022-02-05	31	Hispanic or Latino	6	8.42	7.90	2.6

Mapping

Mapping specifies which variables are used for which aesthetic features. Within `aes()`, we can define the position, color, size, or shape of the plotted values. This layer can be used to highlight underlying patterns in the data [10].

```
# Summarized layer-specific elements
ggolot(data, aes(x, y, group)) +
  [geom_*|stat_*](aes(color, fill, [size|linewidth],
                      [shape|linetype]))

# Worked-Through Example
ggplot(data = df, aes(x = MMWRweek, y = Kernel))
```

Geometry

Geometry functions are the engines that transform data mappings into the desired plot type. They interpret statistical transformations and position adjustments to modulate the graph and can have their own aesthetic mappings defined, which override the defaults set in the `ggplot()` plot object.

```
# Summarized layer-specific elements
ggolot() +
    geom_*(data, aes(x, y, group), position, statistics)

# Worked-Through Example
df |>
  ggplot() +
    geom_line(aes(x = MMWRweek, y = Kernel))
```

Statistics

Statistics layers perform the statistical calculations used by geometric engines, and many can be swapped with their geometric counterparts.

> **Note**
>
> When using a statistically transformed variable for an aesthetic like fill or color, use `after_stat()` around the variable name to map it correctly onto the visual.

```
# Summarized layer-specific elements
ggolot() +
    stat_*(data, aes(x, y, group), geometry, parameters)

# Worked-Through Example
df |>
  ggplot() +
    geom_line(aes(x = MMWRweek, y = Kernel)) +
    stat_smooth(aes(x = MMWRweek, y = `Scaled Positives`),
                geom = "smooth", method = "loess", se = FALSE)
```

Scales

Scales convert aesthetic **Mappings** into plottable values and offer extensive customization, similar to how **Statistics** layers define transformations.

```
# Summarized layer-specific elements
ggolot(data, aes(x, y, group)) + [geom_*|stat_*]() +
  scale_*(parameters)

# Worked-Through Example
df |>
  ggplot(aes(x = MMWRweek, y = Kernel, color = Season)) +
```

```
    geom_line() +
    labs(title, x, y) +
    ylim(0, 100) +
    scale_color_brewer(type = "qual", palette = "Dark2")
```

Facets

Facets split the plot into subgroups of a discrete variable using two functions:

- `facet_grid()`: arranges groups into rows or columns.
- `facet_wrap()`: arranges groups into a rectangular layout, ideal for numerous categories.

```
# Summarized layer-specific elements
ggolot(data, aes(x, y, group)) + [geom_*|stat_*]() +
   facet_*(discrete_variable)

# Worked-Through Example
df |>
  ggplot(aes(…)) +
    geom_line() +
    …
    facet_grid(~Level)
```

Coordinates

Coordinates plot values on a graph within a specified spatial reference system, making the choice of the coordinate system crucial. Scaled or transformed values can look different with incorrect coordinates. For example, plotting an angle in Cartesian rather than polar coordinates [11,12].

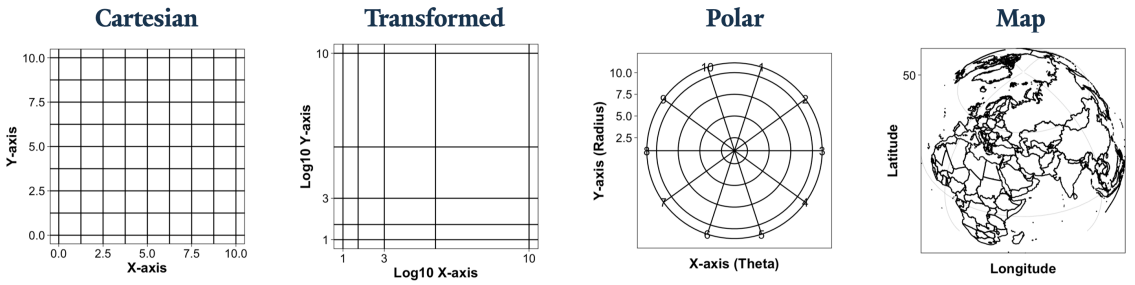

Figure 3: Based on *Data Visualisation: From Theory to Practice Figure 5.5* by James Baglin. Accessed March 22nd, 2025.

```
# Summarized layer-specific elements
ggolot(data, aes(x, y, group)) + [geom_*|stat_*]() +
   coord_*()

# Worked-Through Example
# coord_cartesian() is applied by default.
```

Theme

The **Theme** layer manages non-data aspects of the plot, such as fonts, axis ticks, backgrounds, and legend placement. It enhances visual appeal and style consistency without altering data transformations or geometries.

Code Like a Pro

When generating plots with consistent custom themes, there are two solutions to avoid redundancy [13]:

- `theme_set()`: Set a new global theme, overriding default settings.
- Create a custom theme function: Modify an existing `theme_*()` or create one from scratch.

```
# Summarized layer-specific elements
ggolot(data, aes(x, y, group)) + [geom_*|stat_*]() +
   theme_*()

# Worked-Through Example
df |>
  ggplot(aes(…)) +
    geom_line() +
    …
    theme_linedraw()
```

Glossary

Term	Description
Map Projection	Interpreting the curved surface of the earth into a flat plane for 2D plotting [4,6].
Simple Feature (SF)	Standard vector data produced by the Open Geospatial Consortium (OGC) that translates projection data into plottable polygons [4].

References

1. Wilkinson, L. *The Grammar of Graphics. Springer* (Springer-Verlag, 1999). doi:10.1007/0-387-28695-0.
2. Wickham, H. A layered grammar of graphics. *Journal of Computational and Graphical Statistics* **19**, 3–28 (2010).
3. Pedersen, T. L. ggplot2 workshop part 1. *YouTube* (2020).
4. Wickham, H., Navarro, D. & Pedersen, T. L. *Ggplot2: Elegant Graphics for Data Analysis (3e)*. (Springer, 2010).

5. Wickham, H., Çetinkaya-Rundel, M. & Grolemund, G. *R for Data Science (2e). O'Reilly Media* (O'Reilly Media, 2023).
6. Wickham, H. *et al. Ggplot2: Elegant Graphics for Data Analysis Documentation*. (Springer-Verlag, 2016).
7. Contributors, P. *Data Visualization with Ggplot2: Cheat Sheet*. (Springer-Verlag, 2024).
8. Wickham, H. *et al.* Function references. *ggplot2 Documentation*.
9. Freeman, M. & Ross, J. Chapter 13 the gglot2 library | technical foundations of informatics. (2019).
10. Wickham, H. *et al.* Aesthetic specifications vignette. *ggplot2 Documentation*.
11. Baglin, J. Data visualisation: From theory to practice. (2023).
12. Wickham, H., Navarro, D. & Pedersen, T. L. *15 Coordinate Systems – Ggplot2: Elegant Graphics for Data Analysis (3e)*. (Springer, 2010).
13. Pickens, M. Learning to create custom themes in ggplot2. *RPubs* (2020).